



Europäisches
Patentamt

European
Patent Office

Office européen
des brevets

Bescheinigung

Certificate

Attestation



Die angehefteten Unterla-
gen stimmen mit der
ursprünglich eingereichten
Fassung der auf dem näch-
sten Blatt bezeichneten
europäischen Patentanmel-
dung überein.

The attached documents
are exact copies of the
European patent application
described on the following
page, as originally filed.

Les documents fixés à
cette attestation sont
conformes à la version
initialement déposée de
la demande de brevet
européen spécifiée à la
page suivante.

Patentanmeldung Nr. Patent application No. Demande de brevet n°

00127585.8

**CERTIFIED COPY OF
PRIORITY DOCUMENT**

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

I.L.C. HATTEN-HECKMAN

DEN HAAG, DEN
THE HAGUE, 06/06/01
LA HAYE, LE

This Page Blank (uspto)

307/100-1000
1000-1000-1000
1000-1000-1000



Europäisches
Patentamt

European
Patent Office

Office européen
des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:
Demande n°: 00127585.8

Anmeldetag:
Date of filing:
Date de dépôt: 15/12/00

Anmelder:
Applicant(s):
Demandeur(s):
International Business Machines Corporation
Armonk, NY 10504
UNITED STATES OF AMERICA

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:

Method and system for moving single documents between a document processing system and a document repository

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat:
State:
Pays:

Tag:
Date:
Date:

Aktenzeichen:
File no.
Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:

/

Am Anmeldetag benannte Vertragsstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE/TR
Etats contractants désignés lors du dépôt:

Bemerkungen:
Remarks:
Remarques:

This Page Blank (uspto)

DESCRIPTION

**Method and System for Moving Single Documents between a Document
Processing System and a Document Repository**

Background of the Invention

The invention relates to data processing environments with large document repositories and, more specifically, to a method and system for moving single documents between a document processing system and a document repository.

Known client mailing applications like Lotus™ Notes™ or Microsoft™ Outlook™, up to now, do not provide a standard way of extracting and moving a single note or mail document out of and into a mail database file. This means, that always a whole mail database file must be stored, just for appending a single mail. For this reason, Lotus Notes has already started implementing an XML solution. However, its format as well as its interface are still under evaluation (see <http://www.notes.net/sandbox.nsf/...>).

One drawback of the above described XML approach is that an XML data representation of mails or notes does not preserve all properties of such documents completely.

In addition to that, XML is a readable ASCII-based format. That means, if someone did not have access to an original note, but has the XML representation of a note, one can read the note's content e.g. by browsing it in a regular text editor. Further, converting documents to XML and back means a lot of overhead.

Therefore, a mechanism is needed to extract a document as it is in some compact format without being able to read the document, if it is protected.

Further, when the document is restored, it must be identical to the original one. That is, besides the content, it must also have the same properties.

Summary of the Invention

It is therefore an object of the present invention to provide a method and system for moving a document between a document processing system and a document repository in a compact format, without unnecessary overhead.

It is another object to provide a method and system for moving a document between a document processing system and a document repository in a binary format.

It is another object to provide such a method and system which preserve the properties of a document, in particular structural information and/or security features, when it is moved.

It is yet another object to enable that a restored document is identical to the original document, i.e. besides the content it must also preserve its original unique identifier.

The above objects are achieved by the features of the independent claims. Advantageous embodiments are subject matter of the subclaims.

From a logical point of view, a document processing system allows to read and write single documents, whereas physically those documents are not accessible since they are stored in a sequential binary format. In order to move a single document in its physical representation i.e. the binary format, the invention proposes to use the logical view to obtain all structural information describing the document as well as its content and all meta data. This information will then be used to create an exact copy of that document outside the document processing system.

The invention particularly proposes a native file format to write content such as Lotus Domino notes, that is documents as well as design elements like folders, agents, buttons, etc. to a file. The native file format allows to read content of a document in binary format and write it in a serialized manner to the file. Besides the document content, the file format can store document properties as meta data and therefore those properties are preserved during serialization.

Further, the native format allows that a document restored after serialization is assigned its original unique identifier (ID). In the prior art approaches documents are restored without their original ID and for that reason all document links of remote documents pointing to it would dangle.

Browsing a serialized document that was protected originally is impossible with the proposed native format because it stores information in a binary format which is unknown to users.

When a serialized document is restored, a new, empty document is created. Then it is filled up with the items read from the file. All attachments are restored to their original position. From the original attachment descriptor items, new descriptor items are computed.

Brief Description of the Drawings

In the following, the present invention is described in more detail by way of embodiments from which further features and

advantages of the invention become evident. In the drawings

- Fig. 1 is a graphical representation of a procedure to move a document from a document processing system to a remote document repository according to the invention;
- Fig. 2 shows an exemplary document structure for serialization to a file according to the invention; and
- Fig. 3 shows a scenario for serializing design elements by the example of a folder structure, according to the invention.

Detailed Description of the Drawings

Fig. 1 shows a logical view 100 on a document processing system 101. It depicts an exemplary tree structure 102 including different branches and documents 103 - 106. From a physical point of view 107, the documents 103 - 106 are stored sequentially within a database file 108 as sequential units 109 - 112, indicated by different patterns.

Due to the nature of the sequential file 108, it is impossible to locate a particular document 104 in that sequential file 108. By gaining access to the document 104 via the logical view 100, an exact copy 111 of the physical representation 110 can be extracted, preserving all structural and meta information, and serialized into a binary file 112. This unreadable binary and therefore secure file 112 is then transferred to a document repository 113.

In Lotus Notes, every piece of content or design element is a 'note'. For example, emails, agents, actions (buttons), views, folders and script libraries are notes. Internally, all Notes design elements like documents, views, folders, agents, actions, etc, are special subtypes of the basic object type Note. A note basically consists of a number of items which are multivalued containers for typed data. All information that makes up a note is kept in items, except attachments, which are kept in the database itself.

In Fig. 2 an exemplary document 201 describing a person is shown. The document 201 contains attributes like 'Name' 202, 'Age' 203 and 'Birthday' 204 plus an attachment 205. Both, attributes 202 - 204 and the attachment 205 are represented by items. The proposed native file format reads the binary item content and writes it in a serialized manner to a file 206. Besides the item content, the native format 207 stores meta information like item type and item length. Since attachments are not stored in items, the note's attachments are read from the database and written to the file 206. Further, the attachment's descriptor item and the note's universal ID (UNID) are written to the file 206.

Fig. 3 shows a scenario for serializing design elements by the example of a folder structure 301, according to the invention.

Within Lotus Notes, documents can be organized in folders. When moving Notes documents to a repository 300 it is important to preserve the folder structure in the repository 300 so that it can be restored upon request. As an example, Fig. 3 shows a 'Customer' folder 302 containing customer documents 303, 304.

Generally, within Lotus Notes all structural elements (documents, folders, agents, etc...) are conceptually based on notes. A note is basically a collection of name-value pairs, so called items, describing the element. When moving the entire Notes folder structure to a repository, this is done in three steps:

1. An entry 305 within the repository 300 is created consisting of the descriptive information for the folder 302 (folder name, folder alias, originating database and archiving user). This entry 305 will later contain references to all the subfolders and documents contained in it. The descriptive information is used to retrieve the folder 302 back by name and to ensure that it can only be restored to its originating database;
2. the Notes folder 302 itself will be serialized into a file 306 and the file 306 be transferred 307 to the repository 300. This way all information making up the folder 302 (like columns to display, included actions, formatting and layout information) will be preserved in the repository 300. A reference 308 to the entry 305 done in 1) is created;
3. all documents 303, 304 contained in the Notes folder 302, including the shown document 303, are also serialized 309 and moved 310 to the repository 300, accordingly. For each of them, a reference 311 to the entry 305 made in 1) is created.

If a folder contained subfolders, for each of them steps 1) to 3) would be executed recursively.

Upon request, a complete folder structure including all its subfolders and notes can either be restored by name or - if at least the empty root folder is left in the Notes database - by archive ID to its originating database, i.e. the database it was originally archived from.

When a folder is requested to be restored, the entry in the repository describing that folder is located (either by its ID or through a search on its name in combination with its originating database). Restoring is then done as follows:

1. The repository entry describing the folder in its native format is identified and first restored back to Notes. This will create a new Notes folder or, if a folder with the same name already exists, it will be replaced;
2. each document referenced in the folder in the repository will be restored back to the folder created in 1).

If a subfolder is found upon restore, steps 1) and 2) are repeated recursively.

In a Lotus Notes environment, the following native format properties are preserved:

- serializes a complete note (document, view, folder, agent, action, etc) to a file
- restores a note completely from the Notes native format
- preserves the note's universal id
- preserves attachments and OLE objects
- preserves readers fields, author fields, and all other security fields
- prevents the data from being browsed in an editor because in "secret" native format
- does not preserve signatures as this would be Notes security hole
- platforms: all platforms for which the Notes C and/or C++ API exist

Finally, it is understood that, although the examples above are described in a Lotus Domino environment, the basic concepts can also be used for other document processing systems like Microsoft™ Exchange™.

15-12-2000

EP00127585.8

DE9200001

SPEC

- 6 -

C L A I M S

1. A method for moving a single document between a document processing system and a document repository, comprising the steps of:

Obtaining structural information describing the document structure;

obtaining meta information describing the document properties;

obtaining document content;

creating a physical representation for the document based on the obtained structural information, meta information and document content;

transferring the created physical representation to the document repository.
2. Method according to claim 1, wherein the physical representation for the document is a binary format.
3. Method according to claim 1 or 2, wherein retrieving the document identifier for the moved document and including the document identifier in the created physical representation.
4. Method according to any of the preceding claims, wherein serializing the obtained structural information, meta information and document content into the physical representation for the document.
5. A method for moving a folder and documents contained therein between a document processing system and a document repository, comprising the steps of:

Obtaining structural information describing the folder structure;

obtaining meta information describing the folder properties;

creating a physical representation for the folder based on the obtained structural information and meta information;

transferring the created physical representation to the document repository.
6. Method according to claim 5 for moving a folder including subfolders, comprising the further step of recursively performing the steps in claim 5 for all subfolders and/or sub-subfolders.

7. Method according to claim 5 or 6, wherein the physical representation for the folder is a binary format.
8. Method according to any of claims 5 to 7, wherein serializing the obtained structural information and meta information into the physical representation for the folder.
9. Method according to any of claims 6 to 8, wherein a folder entry contains references to all the subfolders and documents contained in it.
10. A system for moving a single document between a document processing system and a document repository, comprising:

Means for obtaining structural information describing the document structure;

Means for obtaining meta information describing the document properties;

Means for obtaining document content;

Means for creating a physical representation for the document based on the obtained structural information, meta information and document content;

Means for transferring the created physical representation to the document repository.
11. System according to claim 10, comprising means for retrieving the document identifier for the moved document and including the document identifier in the created physical representation.
12. System according to claim 10 or 11, comprising means for serializing the obtained structural information, meta information and document content into the physical representation for the document.
13. A system for moving a folder and documents contained therein between a document processing system and a document repository, comprising:

Means for obtaining structural information describing the folder structure;

Means for obtaining meta information describing the folder properties;

Means for creating a physical representation for the folder based on the obtained structural information and meta information;

means for transferring the created physical representation to the document repository.

14. System according to claim 13 for moving a folder including subfolders, further comprising means for recursively performing the steps in claim 13 for all subfolders and/or sub-subfolders.
15. System according to claim 13 or 14, comprising means for serializing the obtained structural information and meta information into the physical representation for the folder.
16. A data processing program for execution in a data processing system comprising software code portions for performing a method according to any of claims 1 to 9 when said program is run on said computer.
17. A computer program product stored on a computer usable medium, comprising computer readable program means for causing a computer to perform a method according to any of claims 1 to 9 when said program is run on said computer.

A B S T R A C T

Disclosed are a method and system for moving single documents between a document processing system and a document repository.

From a logical point of view, a document processing system allows to read and write single documents. However, physically all documents within a document processing system are usually stored in a sequential binary format and therefore are not accessible in the binary format. In order to move a single document in its physical representation i.e. the binary format, it is proposed to use the logical view to obtain all structural information describing the document as well as its content and all meta data. This information will then be used to create an exact copy of that document outside the document processing system.

The invention particularly proposes a native file format to write content such as Lotus Domino notes, that is documents as well as design elements like folders, agents, buttons, etc. to a file. The native file format allows to read content of a document in binary format and write it in a serialized manner to the file. Besides the document content, the file format can store document properties as meta data and therefore those properties are preserved during serialization.

1 / 3

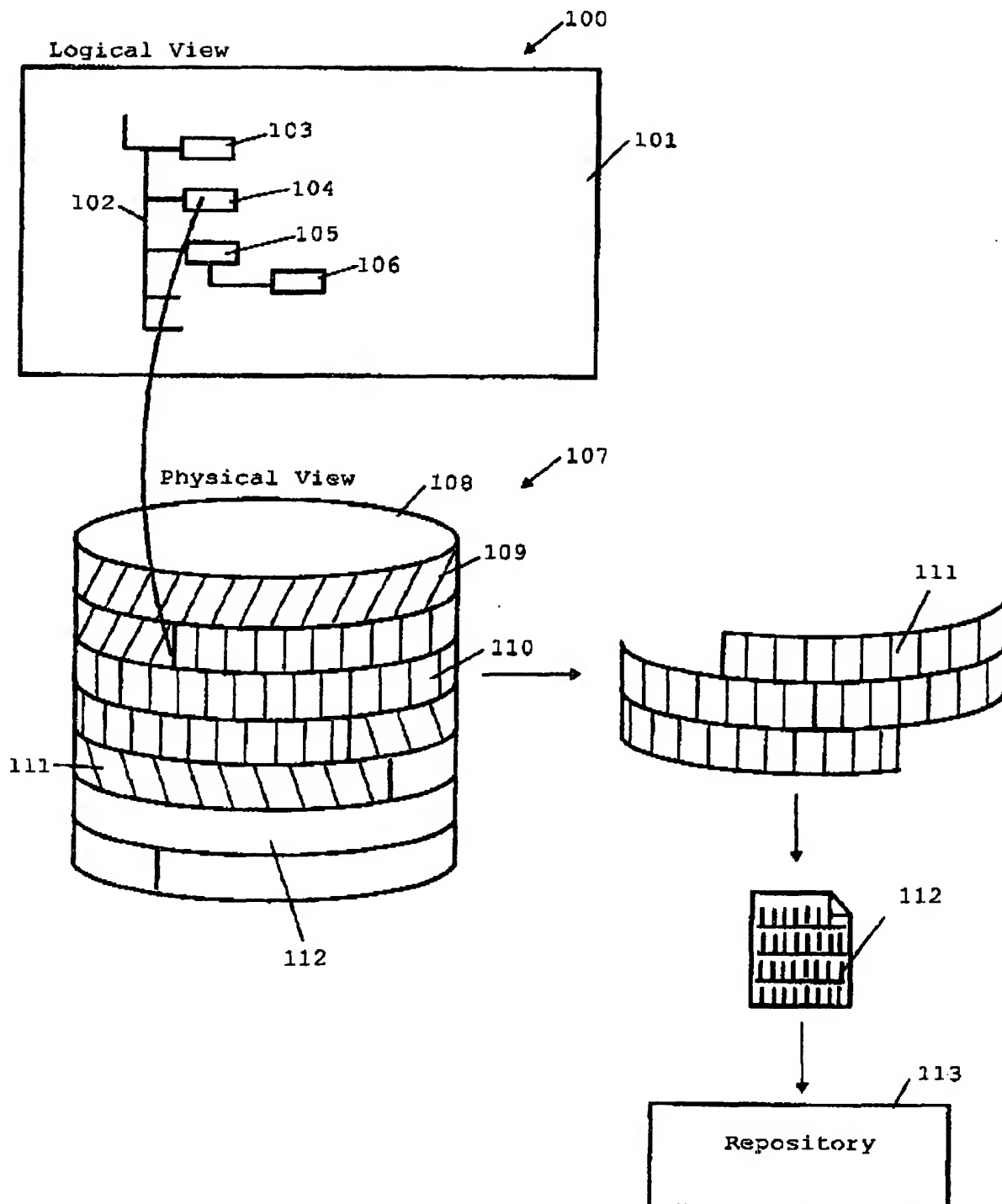


FIG. 1

2 / 3

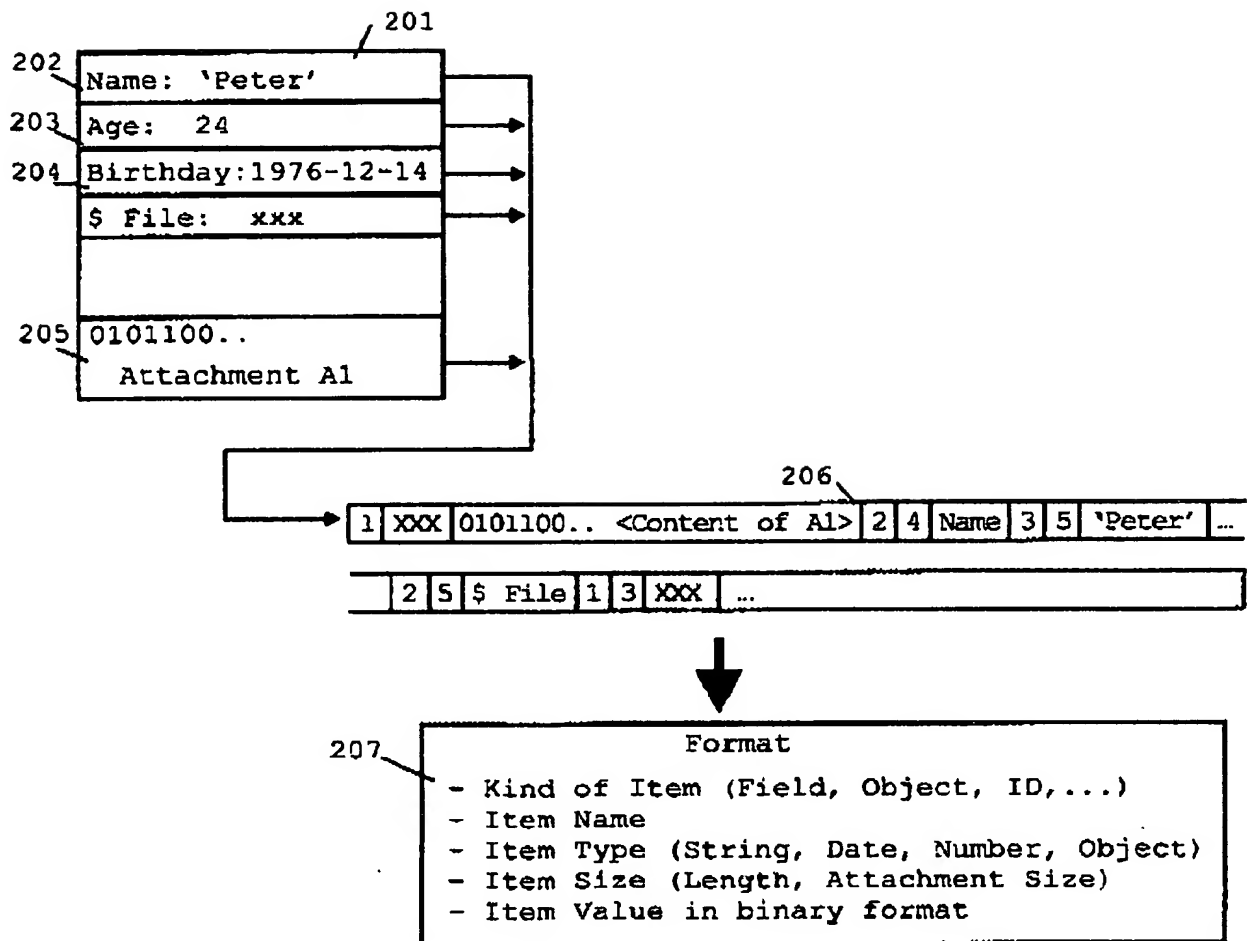


FIG. 2

3 / 3

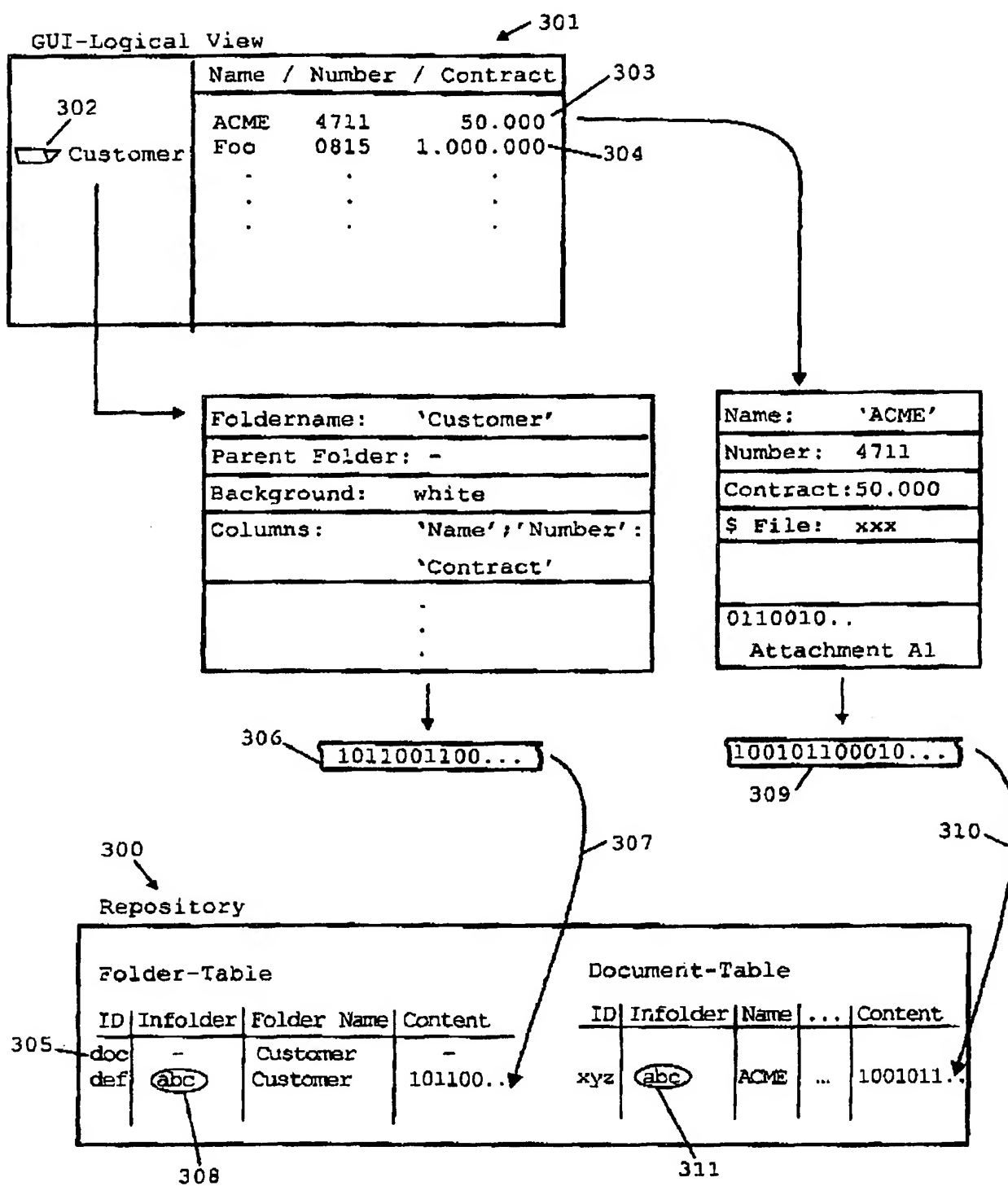


FIG. 3